

**APPENDIX A**

---

## ***GetOFSIOptions***

---

**Method Declaration** GetOFSIOptions(pOFSIOptions As OFSIOptions) As Boolean

**Purpose** The GetOFSIOptions method tells the client what options are supported by this Direct Link Server.

**Parameters** pOFSIOptions (Returned) - OFSIOptions object containing the support status of various server options and how they are supported.

**Return** TRUE if the data is retrieved.  
FALSE if no data is retrieved.

---

## **SetAccounts**

---

**Method Declaration** SetAccounts(pAcctMethod As Integer,  
paAcct1() As object, pfEOF As Boolean) As Boolean

**Purpose** The SetAccounts method defines how the accounts for this Ledger query are defined. There are two methods of getting the accounts:

(1) Account Criteria Provided – If Account Criteria supported, FRx will provide account ranges and wildcards to be used to determine the appropriate accounts and the Direct Link will have to obtain the accounts. If Account Criteria is not supported – FRx will determine the accounts to be included in the ledger query and will provide full account codes to the Direct Link.

(2) Account Types Provided - FRx will provide account types (with wildcards). Accounts with these account types should be included. The Direct Link will have to determine the accounts.

**Parameters** pAcctMethod - This determines the method the accounts will be determined from. Values are:  
frxAcctCriteria (1)  
frxAcctsByAcctTypes (2)

The remaining parameter varies in content based on the pAcctMethod.

If pAcctMethod = frxAcctCriteria,  
paAcct – Array of AcctCriteria class objects

If pAcctMethod = frxAcctsByAcctTypes,  
paAcct – Array of AcctType class objects.

pfEOF - All account criteria will be set at one time with 1 to N calls to SetAccounts. Once all account criteria has been set, FRx will set this flag to TRUE.

**Return** TRUE if processed successfully.  
FALSE if not processed successfully.  
Error raised if error occurs during the setting.

---

**PopulateBalances**

---

**Method Declaration** PopulateBalances() As Boolean

**Purpose** The PopulateBalances method notifies Direct Link Interface (OFIS) that Ledger Criteria, Account Criteria, and Balance Filters are set and GetNextBalance is about to be called.

**Parameters** NA

**Return** TRUE if the data is retrieved.  
FALSE if no data is retrieved.  
Error raised if error retrieved during retrieval.

---

## ***PopulateTransactions***

---

**Method Declaration** PopulateTransactions(pnTranPhases As Long,  
pTranFieldsToFill As Long) As Boolean

**Purpose** The PopulateTransactions method sets up the selection process for retrieving ledger transactions using *GetNextTran*.

**Parameters** pnTranPhases - If the Direct Link (OFSI) option Ledger Unposted Transactions Support is FALSE, this can be ignored. If TRUE and the Direct Link option Unposted Transaction Phases Defined is FALSE, the values for this parameter could be:

2 - Posted Only  
4 - Unposted Only

If both Direct Link Options are TRUE, the values for this parameter are based on BIT-anding of values of the Unposted Phased Defined and 2 for Posted.

pTranFieldsToFill - At a minimum the RowNum, NegSign, AcctCode, ApplyDate, RptngAmt, and TranDesc are returned for each transaction record. To retrieve additional transaction data, we use BIT-anding with the following values for the additional data:

Year - 2  
Period - 4  
BookCode - 8  
ProjCategory - 16  
ProjCode - 32  
OrigAmt - 128  
CurrCode - 256  
ConvRate - 512  
BatchID - 1024  
SequenceNum - 2048  
EntryDate - 4096  
AlternateDate - 8192  
SourceID - 16384  
SourceDocID - 32768  
SourceDocCode - 65536  
SourceBatchDesc - 131072  
AltEntityCode - 262144  
TranType - 524288  
CompCode - 1048576  
AcctID - 2097152  
VDT1 - 4194304  
VDT2 - 8388608  
VDT3 - 16777216  
VDT4 - 33554432

VDT5 - 67108864

So, if you want OrigAmt, CurrCode, and ConvRate, pTranFieldsToFill would be 896 (128 + 256 + 512). If you wanted all the transaction data, pTranFieldsToFill would be 4194304 (all values added together). This parameter is only valid for pDetailLevel 3.

**Return** TRUE if processed successfully.  
FALSE if not processed successfully.  
Error raised if error occurs during the setting.

---

**SetBalanceFilters**

<b>Method Declaration</b>	SetBalanceFilters(paBalFilters() As BalanceFilter, pDetailLevel As Integer, pfSendNoZeroBalAccts As Boolean, pnLedgerSet As Long) As Boolean
---------------------------	--

**Purpose.** The SetBalanceFilters method provides the balance filters. The Balance Filters define the time frame for the balances, book code, currency, debits/credits only, and any additional filtering by account or project.

The balance filters should be analyzed to determine the minimum number of passes necessary to get all the amounts for the ledger query.

**Parameters** paBalFilters - Array of BalanceFilter class objects.

**pDetailLevel** - Defines the detail level for balance data and if transactions data is requested.

- 1 - Sum all accounts balances into one balance record.
- 2 - Return all account balance records.
- 3 - Return all account balance records and transactions detail will also be returned.

**pfSendNoZeroBalAccts** - If TRUE, accounts where all the Balance Amounts are zero will not be returned. If FALSE, for each account determined in *SetAccounts* a balance object is returned via *GetNextBalance*.

**pnLedgerSet** - Identifies which ledger set within ledger criteria these balance filters apply to.

**Return** TRUE if processed successfully.  
FALSE if not processed successfully.  
Error raised if error occurs during the setting.